

# Statistical performance assessment of supervised machine learning algorithms for intrusion detection system

Hassan A. Afolabi, Abdurazzag A. Aburas

Discipline of Computer Engineering, School of Engineering, University of Kwazulu-Natal, Durban, South Africa

## Article Info

### Article history:

Received Sep 27, 2022

Revised Feb 6, 2023

Accepted Mar 10, 2023

### Keywords:

Ensemble learning  
Internet of things  
Intrusion detection  
Performance analysis  
Significance test

## ABSTRACT

Several studies have shown that an ensemble classifier's effectiveness is directly correlated with the diversity of its members. However, the algorithms used to build the base learners are one of the issues encountered when using a stacking ensemble. Given the number of options, choosing the best ones might be challenging. In this study, we selected some of the most extensively applied supervised machine learning algorithms and performed a performance evaluation in terms of well-known metrics and validation methods using two internet of things (IoT) intrusion detection datasets, namely network-based anomaly internet of things (N-BaIoT) and internet of things intrusion detection dataset (IoTID20). Friedman and Dunn's tests are used to statistically examine the significant differences between the classifier groups. The goal of this study is to encourage security researchers to develop an intrusion detection system (IDS) using ensemble learning and to propose an appropriate method for selecting diverse base classifiers for a stacking-type ensemble. The performance results indicate that adaptive boosting, and gradient boosting (GB), gradient boosting machines (GBM), light gradient boosting machines (LGBM), extreme gradient boosting (XGB) and deep neural network (DNN) classifiers exhibit better trade-off between the performance parameters and classification time making them ideal choices for developing anomaly-based IDSs.

*This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.*



## Corresponding Author:

Abdurazzag A. Aburas

School of Electrical, Electronic and Computer Engineering, University of Kwazulu-Natal

Durban, South Africa

Email: aburasa@ukzn.ac.za

## 1. INTRODUCTION

Businesses, manufacturing sectors, and financial institutions are all becoming more and more reliant on technology. The risk of cyberattacks is therefore extremely high. Therefore, protecting these devices is one of the main issues facing researchers today [1]. Intrusion detection is a topic that is the subject of extensive research globally [2], [3]. Based on the mechanism used for detection, intrusion detection systems (IDSs) are divided into three classes: signature, anomaly, and specification-based. This is better explained with Figure 1 which depicts an intrusion detection system classified by detection strategy, deployment, architecture, and detection behaviour or responses.

Anomaly-based IDS [4] is preferable to signature-based and specification-based IDS due to its ability to detect novel threats. The problems of misclassification of intrusions in intrusion detection systems can be solved by a supervised learning algorithm [5]. Many supervised machines learning methods, including individual classifiers and ensemble classifiers, have been used to develop anomaly detection systems. Even while combining several classifiers has helped machine learning research develop over the past decade, selecting the appropriate ones to combine can be challenging, especially when employing an ensemble of the stacking type.

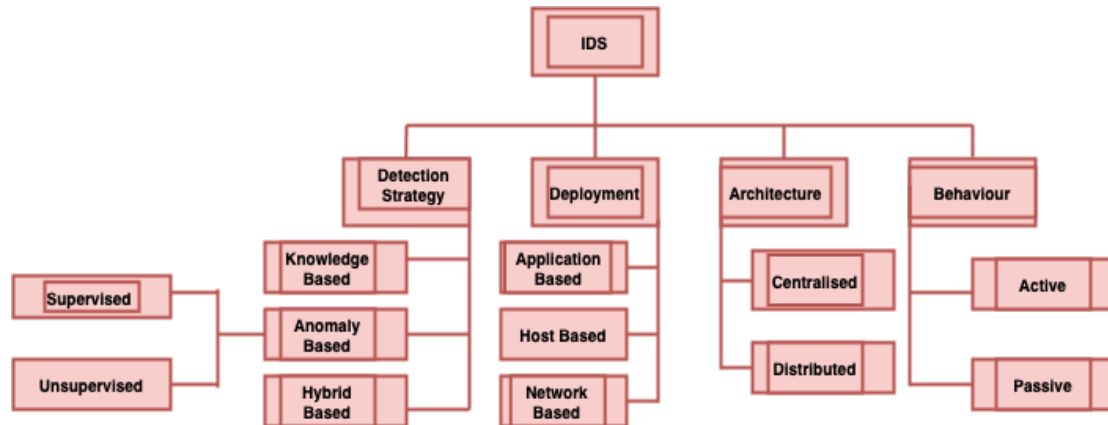


Figure 1. Taxonomy of IDS

It is necessary to compare the performances of the best machine learning (ML) classifiers in ML studies. This is a major challenge, especially when compared across various datasets [6]. An algorithm could perform well on one dataset while failing to get the same results on another. This can be due to the existence of outliers, feature distribution, or algorithm properties. As a result, comparing several algorithms to one another becomes rather challenging.

Several studies on the performance evaluation of ML classifiers, particularly in network attack detection, have been conducted. Subbiah *et al.* [5] presented a novel framework for intrusion detection that is enabled by Boruta feature selection with grid search random forest (BFS-GSRF) algorithm. The proposed work was evaluated on the knowledge discovery and data mining (NSL-KDD) dataset and its performance were compared to linear discriminant analysis (LDA) and classification and regression tree (CART). According to the results obtained in their study, the proposed BFS-GSRF outperforms LDA, CART and other existing algorithms with an accuracy of 99% in detecting attacks. Studies in [7] investigated the comparative study of several ML methods used in IDS for a variety of applications such as big data, fog computing, internet of things (IoT), smart cities, and 5G networks. Furthermore, they classify intrusions using classifiers such as CART, LDA, and RF, and implemented on the knowledge discovery and data mining tools competition (KDD-CUP'99) dataset, and their efficiency was measured and compared to recent researches. Zaman and Lung [8] used ensemble methods, fuzzy c-means, naive bayes, radial basis function and support vector machine (SVM) to build an IDS using the Kyoto 2006+ dataset. They obtained promising results in terms of accuracy with ensemble methods reaching 96.72%.

A gradient boosted machine (GBM) is a suggested detection engine for an anomaly-based IDS by [9] using various datasets which are general packet radio service (GPRS), NSL-KDD and University of New South Wales network-based attacks (UNSW-NB15), the proposed IDS's performance is assessed with hold-out and cross-fold techniques, and the optimal GBM parameters are obtained using grid search. The proposed method outperformed fuzzy classifiers and tree-based ensembles in terms of all the metrics considered. Kilincer *et al.* [10] conducted a thorough literature review in 2021 to compare the performance of SVM, k-nearest neighbors and decision tree (DT). The communications security establishment and the canadian institute for cybersecurity intrusion detection evaluation dataset (CSE-CIC-IDS2018), UNSW-NB15, information security centre of excellence intrusion detection evaluation dataset (ISCX-2012), NSL-KDD, and the cyber intrusion detection system dataset 001 (CIDDS-001) datasets were all used for the comparative analysis. Except for the UNSW-NB15 dataset, the study found that the accuracy of the models varied from 95% to 100%. DT consistently outperformed all other implemented models, irrespective of dataset. The ability to detect unknown threats is also a concern when evaluating the performance of the IDS. Hindy *et al.* [11] studied the effectiveness of ML-based IDS in detecting unknown attacks in 2020. The study proposed an intrusion detection system (IDS) that could detect zero-day threats with high recall rates while having a low false positive rate. In addition, to compare with the proposed model, they implemented a one-class SVM. The canadian institute of cybersecurity intrusion detection evaluation dataset (CICIDS2017) and the NSL-KDD dataset were used for model training and evaluation in this study. To achieve the zero-day attack detection setting, only normal traffic was used when training the model, and all attack samples were used to simulate zero-day attacks. The study found that both models had a low false positive rate when it came to detecting zero-day attacks. Pranto *et al.* [12] demonstrated a method to classify NSL-KDD dataset as normal or malicious using several machine learning techniques such as k-nearest neighbor, decision tree, Naïve Bayes, logistic regression, random forest, and their ensemble approach. They obtained a highest accuracy of 99.5% with a 0.6% false alarm rate. Making a choice

about which algorithm is superior to others is therefore difficult. Statistical validation of the performance outcomes is required to address this problem.

The performance of ML classifiers for the design of an IDS is evaluated in this work. The accuracy, precision, recall, and f-score of classifiers such as logistic regression (LR), stochastic gradient descent classifier (SGDC), deep neural network (DNN), random forest (RF), adaptive boosting (AB), extreme gradient boosting (XGB), gradient boosting machine (GBM), and extra tree classifier (ETC), are all measured. All classifiers, excluding DNN, have their hyper-parameters tuned via random search [13]. Using well-known statistical tests, the major differences between classifiers are statistically examined. Our major contributions are summarized. i) Using well-known validation methods, the performance of various ML classifiers on the network-based anomaly internet of things (N-BaIoT) and internet of things Intrusion detection dataset (IoTID20) datasets is evaluated; ii) Statistical evaluation of performance results are carried out with the widely used Friedman and Dunn's post-hoc tests. The significance of classifier was conducted with Friedman's test and the Dunn's test was used for pairwise comparison between the ML classifiers; iii) Classifiers that exhibit better trade-off between the metrics considered and evaluation time are recommended for the design of IoT-specific anomaly-based IDS. The remainder of the paper is structured by briefly discussing the materials and methods adopted in our work in section 2, section 3 discusses the experimental setup and classifier performance. Additionally, it discussed the statistical evaluations that were carried out. The paper is summarized and concluded in section 4.

## 2. MATERIALS AND METHODS

This section describes the materials and methods used in this study. Firstly, we discussed about the learning techniques and their characteristics. Then, we discussed about the datasets that was used, experimental tools and their configurations.

### 2.1. Classification algorithms

The classification algorithms used in our study are described briefly in this section. To start with, we provided a general definition of ML classifiers and DL concepts. Furthermore, it introduces Ensemble learning approaches and explains how they can be used to boost classification performance.

#### 2.1.1. Logistic regression

Logistic regression is a supervised machine learning technique that works with categorical dependent variables. Its aim is to assign data to the correct classes based on their correlation. Logistic regression equation can be derived by substituting (1) in (2). Where  $\beta_0, \beta_1 \dots \beta_n$  are the regression parameters,  $x_1, x_2 \dots x_n$  are values of the predictors.

$$p = \frac{1}{1+e^{-y}} \quad (1)$$

$$\ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \dots \beta_n x_n \quad (2)$$

#### 2.1.2. Stochastic gradient descent

They are a variation of gradient descent methods that address computational time issues. Stochastic gradient descent (SGD) computes the gradient of a selected subset of observations rather than all of them. A few machine learning libraries may cause confusion between the two concepts. In scikit-learn, for example, a model called SGD classifier may prompt a user to believe that SGD is a classifier. That is, however, an SGD-optimized linear classifier.

#### 2.1.3. Deep learning

Deep learning enables scalable training of nonlinear models on huge datasets and succeeds at generalizing new examples when the input data is complex and has high dimensionality [14]. Multiple-layered artificial neural networks (ANN) are used for the learning process. Let's assume that  $(x_1$  and  $x_2)$  are the input vector, and the neuron's output is given by,

$$y = \sigma(x_1 w_1 + x_2 w_2 + b) \quad (3)$$

where  $y$  denotes the output,  $\sigma$  = activation function,  $w_1$  and  $w_2$  denote connection. Weights, and  $b$  denotes bias. An activation function's role is to introduce nonlinearity and enable the model to learn complex nonlinear functions.

#### 2.1.4. Extreme gradient boosting (XGBoost)

Extreme gradient boosting (XGBoost) is a boosting method that is part of the ensemble-based approach that uses decision trees to build a model. Chen and Guestrin first presented the XGBoost algorithm at the association for computing machinery (ACM) special interest group on knowledge discovery and data mining (SIGKDD) conference in 2016, and it has since had a significant impact on the machine learning community globally, from winning Kaggle contests to solving significant problems [15]. XGBoost algorithm performs best with medium to small dataset size and it is approximately 10 times quicker than the existing techniques on a single platform, removing time consumption issues, particularly during network data pre-processing.

#### 2.1.5. Light gradient boosting machines (LGBM)

LGBM is a decision tree-based high-performance gradient boosting system. LGBM filters samples for finding divided values using gradient-based one-sided sampling (GOSS). It can be used for classification, ranking, and several other machine learning tasks. Based on the best fit, LGBM splits the tree leaf\_wise. This approach can significantly reduce loss more than other boosting techniques.

#### 2.1.6. Gradient boosting machine (GBM)

GBM is a part of the "ensembles" method family that attempt to enhance how well decision trees perform [16]. It sequentially aggregates weak classifiers and allows them to optimise any differential loss function to produce a robust prediction model. This is similar to other boosting techniques. In order to reduce prediction errors, each current learner (Tree) is dependent on the predictions of earlier learners.

#### 2.1.7. Extremely randomized tree (ET)

This classifier is occasionally referred to as extra trees classifier (ETC) [17]. It is a tree induction approach that can be used for supervised classification and regression. It creates a group of DTs that haven't been pruned. One of the essential steps in ETC is to select the features and the cut-point at random, without respect to the dependent variable. At each tree node, this process is repeated by choosing one or more attributes to use in making the best decision.

#### 2.1.8. Random forest (RF)

Random forest is a group of trees which make predictions individually on a particular input  $x_{in}$  [18]. Each predictor is based on a random group of variables which are taken from the same distribution independently. The basic idea behind RF is that if you use more predictors, you might be able to make more accurate predictions and avoid the problem of overfitting. In RF, each tree grows to a certain size without being pruned. When a lot of trees are formed, they make predictions by voting for the most popular class at input  $x_{in}$ .

#### 2.1.9. Adaptive boosting

Adaptive booting is an adaptable meta-estimator that learns initial training weights from the original dataset [19]. These weights are used to feed more instances into the classifier based on the ones that were misclassified. The subsequent classifiers modify the weights of instances that have been classified, i.e., complex cases. By boosting weak learners, adaptive boosting improves the performance of learning algorithms and enables the final model to converge to strong learners. In (4) represents a boosting approach in which  $x$  is equal to an input object and  $c_p$  is equal to a weak learner.

$$C_P(x) = \sum_{p=1}^P c_p(x). \quad (4)$$

### 2.2. Datasets

Dataset availability is important in the domain of intrusion detection systems because ML problems are heavily reliant on data. Additionally, the quality of the datasets available is also important because the higher the data quality, the better the algorithms perform. In this study, we've chosen two groups of some of the most extensively applied and popular supervised machine learning algorithms and performed a performance evaluation in terms of well-known metrics and validation methods using two internet of things (IoT) intrusion detection datasets, namely N-BaIoT and IoTID20.

#### 2.2.1. N-BaIoT dataset

The N-BaIoT dataset was developed in 2018 in order to address the limitations of freely accessible botnet datasets, specifically for IoT. The IoT sensor traffic was collected in a local network using Wireshark in the central switch and infected by genuine botnets from two families, namely mirai and bashlite, which are two of the most common IoT-based botnets and have already shown their malicious abilities [20], [21]. The

N-BaIoT dataset includes files that each have 115 independent features designed statistically that is derived from the raw network packet (pcap) files as well as a class label. All 115 features are self-explanatory. The class instance occurrence in the N-BaIoT dataset is shown in Table 1.

Table 1. NBaIoT category label distribution

Attack type	No of samples
Benign	13,113
Mirai	101,409
Gafgyt	115,307

### 2.2.2. IoTID20 dataset

IoTID20 dataset is a new dataset proposed by authors in [22] and can be assessed in [23]. IoTID20 dataset is originally created by implementing two basic smart home devices namely the SKT NUGU (NU100) and the EZVIZ Wi-Fi camera (C2C Mini O Plus 1080P) connected to a smart home wi-fi-router. All other devices involved such as laptops, tablets, and smartphones are connected to the same wireless network. The SKT NGU and EZVIZ Wi-Fi camera are IoT victim devices and all other devices in the testbed are the attacking devices. The number of normal and attack instances in IoTID20 dataset are described in Table 2.

Table 2. IoTID20 category label distribution

Class label	No of instances
Benign	40,073
Denial of service attack	59,391
Mirai	415,677
Man in the middle attack	35,377
Scan	75,265

### 2.3. Performance metrics

The performance of the classifiers is evaluated on the test set of the experimental dataset using metrics such as accuracy, precision, recall and  $F_{measure}$ . Accuracy is defined as the ratio of correctly predicted classes, precision is the proportion of positive instances predicted that are actually positive, recall, also referred to as the sensitivity or true positive rate (TPR) or attack detection rate (in intrusion detection problems), is the proportion of actual positive instances that are accurately predicted as positive,  $F_{measure}$  is the harmonic mean of the precision and the recall, it is mostly used when comparing models. The mathematical expressions for computing the metrics are given in (5)-(8),

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (5)$$

$$precision = \frac{TP}{TP+FP} \quad (6)$$

$$recall = TPR = DR = \frac{TP}{TP+FN} \quad (7)$$

$$F_{measure} = 2 \times \frac{precision \times recall}{precision + recall} \quad (8)$$

## 3. EXPERIMENTAL SETUP, RESULTS, AND DISCUSSION

### 3.1. Experimental setup

The algorithms used to generate the pool of classifiers for this experiment are implemented in Jupyter notebook version 6.0.3 of the Anaconda platform with Python 3, Keras, and TensorFlow using the automatic machine learning (AutoML) framework. To produce the best-performing models, it utilizes grid search and model parameter tuning during the classifier generation stage. The random grid search was implemented in the scikit-learn package using Python 3. We utilized the technique proposed in [24] to handle class imbalance problems in the datasets. To determine how well these classifiers performed overall, we ran repeated experiments using hold-out and k-fold cross-validation methods.

To create a train and test set for hold-out validation, we divided the sample dataset into a 67:33 ratio (67% training instances and 33% testing instances). Similarly, the value of k for k-fold cross-validation is

considered to be 10. All performance results given in this section are the weighted average of outputs from 10 iterations of every repeated validation approach to avoid bias. The process flow of the methodology is illustrated in Figure 2. The classifier evaluation time is measured during the testing phase (It is the time measured from when the classification process starts until the classification process stops). Using the following command,

```
start_time = time.time()
```

```
y_pred = clf.predict(X_test)
```

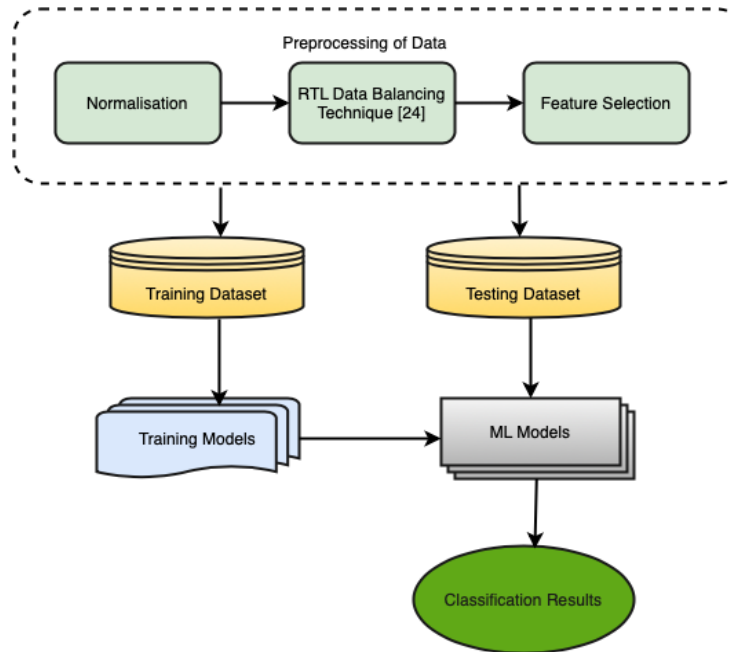


Figure 2. Process flow of the methodology

### 3.2. Results and discussion

#### 3.2.1. Performance

The hyper-tuned parameter configurations of the various boosting algorithms are generated from randomized grid search described in the previous sections are first presented in this section. The best hyper-parameters for GBM are 500 estimators, maximum depth of the construction the tree is 3, a learning rate of 0.1 and 100 samples is the minimum required for split. The hyper-tuned parameters for ETC are 1,788 estimators, maximum value of tree depth is 10, minimum split for sample size is 5,  $\log_{10}2$  is the number of features used for best split and gini with no bootstrapping is the criterion considered. AB's optimal parameters are 50 estimators and a learning rate of 0.1. XGBoost's best hyperparameters are 600 estimators, gini of 0.3, maximum depth of 6, minimum child weight of 5, `colsample_bytree` of 0.6 and learning rate of 0.05.

Additionally, the parameters of the deep neural network were adjusted by passing a new training set into the network. The two network parameters that were tuned in this study is the learning rate and the number of iterations (epochs). As candidate parameters for the model, a range of learning rate which are [0.10, 0.010, 0.0010, and 0.00010] were selected. Measurement standard is the model's  $F_{\text{measure}}$  on the verification set. The appropriate epoch numbers are obtained during the training of the model as the loss function value changes. It is concluded that training should end when peak performance is achieved. Otherwise, the performance value could deteriorate. 0.001 was selected as the learning rate. Similarly, the number of epochs was set to 50 because the network stabilizes at the 50<sup>th</sup> iteration. Other parameters of the DNN model are two hidden layers: layer one has 64 neurons and layer two has 32 neurons. Rectified linear activation function (ReLU) is the activation function utilized at the hidden layers, while sigmoid was employed the output layer. The optimization algorithm used is adaptive moment optimizer (Adam) [25]. Although parameter tuning was done using only NbaIoT dataset, we used same network parameters for both datasets. The model configurations are given in the Table 3.

Table 3. Description of DNN algorithm's hyperparameters

Description	Value
Hidden layers	2
Neurons at hidden at neurons 1 & 2 respectively	64, 32
Output	1
Learning rate	0.001
Activation functions	ReLu, Sigmoid
Optimization	Adam optimizer
Loss Function	Binary cross entropy
Epoch	50

Secondly, the study examined the hold-out validation performance results. Figures 3 and 4 illustrates the results for the two experiments conducted for hold-out validation on N-BaIoT and IoTID20 datasets respectively. The mean value of the performance achieved with hold-out validation across both datasets is shown in Figure 5 which illustrates that classifier outperform each other in terms of various metrics.

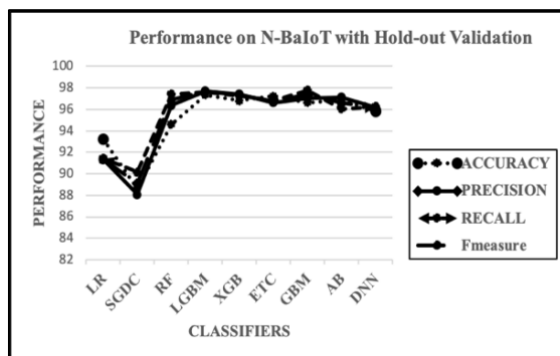


Figure 3. Results on N-BaIoT dataset

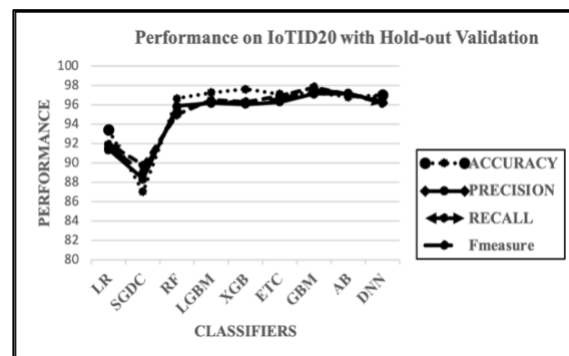


Figure 4. Results on IoTID20 dataset

For the ensemble algorithms, LGBM performs better than other ensembles in terms of accuracy (97.31%), while AB performs better in terms of precision (97.18%), GBM outperforms the others in terms of recall (97.82%), and F-measure outperforms the others (97.49%). Furthermore, when only single classifiers are considered, DNN outperforms LR and SGDC in terms of all metrics considered (96.42%, 96.23%, 96.25%, and 96.24%) for accuracy, precision, recall, and F-measure, respectively. SGDC, on the other hand, achieves the lowest values in terms of accuracy, precision, recall, and F-measure, with 87.84%, 88.21%, 89.99%, and 89.09%, respectively, among all classifiers considered.

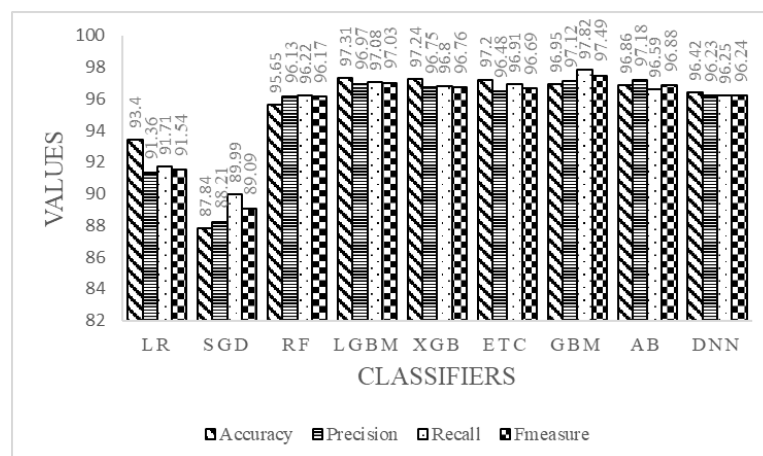


Figure 5. The average of each classifier's metrics with hold-out validation

Figures 6 and 7 illustrates the results for the experiments conducted for 10f validation on N-BaIoT and IoTID20 datasets respectively. The average values of all notable metrics obtained with 10f validation across both datasets are shown in Figure 8. In comparison to classifier performances with hold-out validation, all utilized classifiers perform better with 10f validation. This is a result of sampling's effect, which selects random occurrences and results in unsatisfactory classification.

The adoption of 10f validation rather than hold-out validation is encouraged by this phenomenon. The 10f validation results indicate that each classifier has a promising level of performance. DNN, on the other hand, outperforms all other methods in terms of accuracy (98.76%). AB has the highest average precision measure (98.30%). With recall and F-measure scores of 98.83% and 98.53%, respectively, GBM performs best. Like the hold-out validation, SGDC still yields the lowest results when all metrics are taken into consideration.

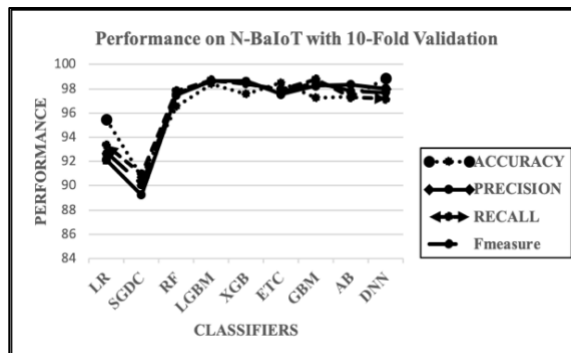


Figure 6. 10f Results on N-BaIoT dataset

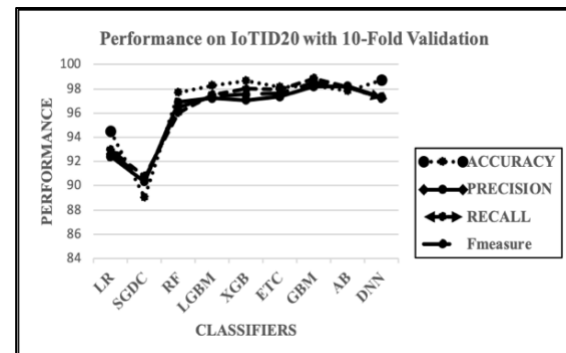


Figure 7. 10f Results on IoTID20 dataset

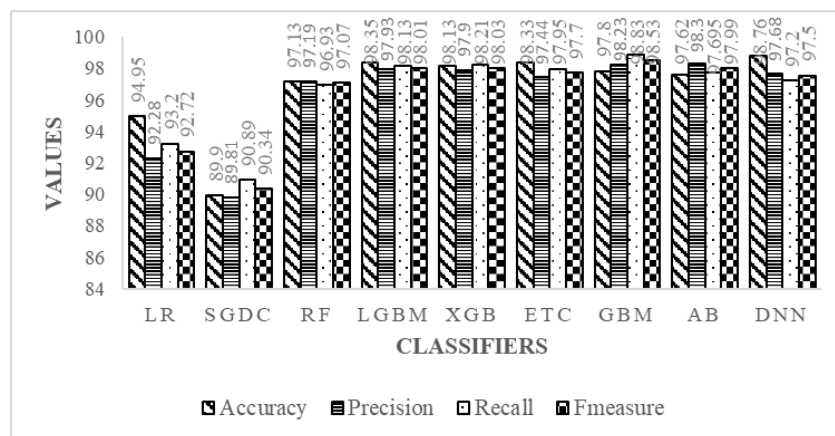


Figure 8. The average of each classifier's metrics with 10-fold validation

Classifier test times are listed in Table 4. Given that resource utilization is a key criterion for devices with limited resources, it is crucial to take a classifier's evaluation time into consideration because it assists in creating a suitable trade-off between a classifier's classification performance and resource utilization. On the N-BaIoT and IoTID20 datasets respectively, LGBM training takes about 11 and 3 seconds. For both datasets, DNN requires the most time for model evaluation. The test time of each classifier is generated solely for 10-fold validation.

Table 4. Classifier test time (Seconds) across both datasets

Dataset	LR	SGDC	RF	LGBM	XGB	ETC	GBM	AB	DNN
N-BaIoT	50	43	14	11	27	15	20	29	106
IoTID20	37	28	8	3	9	7	13	17	97



### 3.2.2. Statistical assessment

The Friedman and Dunn post-hoc test is used to statistically analyze the performance results. The Friedman test can help determine whether one classifier performs better than the others by a significant margin across all datasets. If one of these classifiers is discovered, the pairwise multiple comparisons of Dunn's post-hoc test are performed to identify where that difference lies. Friedman test was chosen primarily because it is the most potent statistical test when more than five entities are being compared [26]. It is essential to undertake post-hoc tests, as proposed in [6] to identify differences in performance between classifiers.

Due to the application of IoT in delicate industries like healthcare and financial institutions, a very low significance level set at ( $\alpha=0.05$ ) was chosen for this experiment. It is crucial to consider even the smallest difference in classifier performance. The null hypothesis ( $H_0$ ) and the alternative hypothesis ( $H_A$ ) are described in Table 2. The values of  $d$  and  $k$  for the numbers of datasets and classifiers considered in this experimental study is 2 and 9 respectively. For  $\alpha = \{0.05\}$  the values of degree of freedom  $f_1$  and  $f_2$  is 8, obtained from the formula,  $f_1 = k - 1$  and  $f_2 = (d - 1)(k - 1)$ . The summary of the test hypothesis and the friedmann's test statistics for hold-out validation are given in Tables 5 and 6 respectively.

Table 5. Hypothesis test summary

$H_0$	$H_A$	Test	sig. level	Asymp. Sig	Decision
The distributions of the classifiers are the same.	At least one Classifier different significantly from the rest.	Related-Samples Friedman's Two-Way Analysis of Variance by Ranks	0.05	Displayed	Reject the null hypothesis.

Table 6. Friedmann's test statistics for holdout validation

Metrics	Score
N	4
Chi-square	28.533
df	8
Asymp. Sig	.000

According to the results presented in Tables 5 and 6, the classifiers' performance is significantly different across all of the evaluated assessment metrics. As a result, it may be said that at least one classifier performs much better than the others. Therefore, the alternative hypothesis  $H_A$  is accepted, whereas the null hypothesis  $H_0$  is rejected. The mean ranks of each classifier used for hold-out validation are shown in the Table 7.

Table 7. Mean ranks of Friedman test for hold-out validation

Classifier	Ranks
LR	2.00
SGDC	1.00
RF	3.00
LGBM	8.00
XGB	6.50
ETC	6.00
GBM	8.00
AB	6.50
DNN	4.00

Dunn's post-hoc test is used to determine which classifier pairs perform significantly differently. All pairwise comparisons'  $p$  values are checked against the considered significance level of 0.05 for this purpose. Assuming that  $C_1$  and  $C_2$  are the test classifiers, the results of Dunn's test (Pairwise comparison) for hold-out validation are shown in Table 8.

According to Table 8, the classifiers for SGDC and LR vs (ETC, XGB, AB, LGBM, and GBM) are statistically and significantly different ( $p < 0.05$ ), whereas the classifiers for RF-LGBM, RF-GBM, DNN-LGBM, and DNN-GBM pairs are less significant. Please take note that the pairs of classifiers that are red-shaded are not significantly different. Tables 9 and 10 shows the friedman test statistics and the mean ranks of each classifier respectively for 10-fold validation results.

Table 8. Dunn's pairwise comparison for hold-out validation

$C_1 - C_2$	P-value	Decision	$C_1 - C_2$	P-value	Decision
SGDC-LR	0.606	A	LR-DNN	0.302	A
SGDC-RF	0.302	A	LR-ETC	0.039	R
SGDC-DNN	0.121	A	LR-XGB	0.020	R
SGDC-ETC	0.010	R	LR-AB	0.020	R
SGDC-XGB	0.005	R	LR-LGBM	0.002	R
SGDC-AB	0.005	R	LR-GBM	0.002	R
SGDC-LGBM	0.000	R	RF-DNN	0.606	A
SGDC-GBM	0.000	R	RF-ETC	0.121	A
LR-RF	0.606	A	RF-XGB	0.071	A
RF-AB	0.071	A	ETC-AB	0.796	A
RF-LGBM	0.010	R	ETC-LGBM	0.302	A
RF-GBM	0.010	R	ETC-GBM	0.302	A
DNN-ETC	0.302	A	XGB-LGBM	0.439	A
DNN-XGB	0.197	A	AB-LGBM	0.439	A
DNN-AB	0.197	A	XGB-AB	1.000	A
DNN-LGBM	0.039	R	XGB-GBM	0.439	A
DNN-GBM	0.039	R	AB-GBM	0.439	A
ETC-XGB	0.796	A	LGBM-GBM	1.000	A

Table 9. Friedmann's test statistics for 10-fold validation

Metrics	Score
N	4
Chi-square	25.133
df	8
Asymp. Sig.	.001

Table 10. Mean ranks of Friedman test for 10-fold validation

Classifier	Ranks
LR	2.00
SGDC	1.00
RF	3.00
LGBM	7.25
XGB	7.00
ETC	5.50
GBM	7.75
AB	6.00
DNN	5.50

To determine which classifier performs statistically differently for 10-fold validation, Dunn's post-hoc test is used. The outcomes are displayed in Table 11. The classifiers are statistically and significantly different ( $p < 0.05$ ) when comparing the classifiers SGDC to DNN, ETC, XGB, AB, LGBM and GBM, LR to XGB, AB, LGBM and GBM, and RF-LGBM, RF-GBM, and RF-XGB pairs. Please note the red-shaded to be the non-significantly different pair of classifiers for 10-fold validation.

Table 11. Dunn's pairwise comparison for 10-fold validation

$C_1 - C_2$	P-value	DECISION	$C_1 - C_2$	P-value	DECISION
SGDC-LR	0.606	A	LR-DNN	0.071	A
SGDC-RF	0.302	A	LR-ETC	0.071	A
SGDC-DNN	0.020	R	LR-XGB	0.010	R
SGDC-ETC	0.020	R	LR-AB	0.039	R
SGDC-XGB	0.002	R	LR-LGBM	0.007	R
SGDC-AB	0.010	R	LR-GBM	0.003	R
SGDC-LGBM	0.001	R	RF-DNN	0.197	A
SGDC-GBM	0.000	R	RF-ETC	0.197	A
LR-RF	0.606	A	RF-XGB	0.039	R
RF-AB	0.121	A	ETC-AB	0.796	A
RF-LGBM	0.028	R	ETC-LGBM	0.366	A
RF-GBM	0.014	R	ETC-GBM	0.245	A
DNN-ETC	1.000	A	XGB-LGBM	0.897	A
DNN-XGB	0.439	A	AB-LGBM	0.519	A
DNN-AB	0.796	A	XGB-AB	1.000	A
DNN-LGBM	0.366	A	XGB-GBM	0.699	A
DNN-GBM	0.245	A	AB-GBM	0.366	A

ETC-XGB	0.439	A	LGBM-GBM	0.796	A
---------	-------	---	----------	-------	---

#### 4. SUMMARY AND CONCLUSION

An investigation into anomaly-based IDS techniques that can be used to secure IoT was conducted in this research. The evaluation of the effectiveness of supervised machine learning algorithms such as RF, AB, LGBM, GBM, ETC, XGB, SGDC, LR, and DNN are given special attention. The best parameters for the classifiers are found using a random search approach. Benchmark datasets like NbaIoT and IoTID20 are used to evaluate classifier performance. Accuracy, precision, recall, and Fmeasure are used to evaluate the effectiveness of each classifier. The training time of all classifiers is measured during the training phase. Additionally, Friedman and Dunn's post-hoc tests are used in the statistical analysis of performance measurements to detect significant differences amongst classifiers.

The performance results and statistical analysis indicates that DNN, Adaptive Boosting, and Gradient Boosting Machines (GBM, LGBM, XGB) classifiers exhibit the appropriate trade-off between performance metrics and evaluation time of 106s, 29s, 20s, 11s and 27s respectively for N-BaIoT dataset, and 97s, 17s, 13s, 3s and 9s respectively for IoTID20 dataset. These results made them the ideal choices for developing anomaly-based IDS specifically for IoT. The design of an IDS based on a stacking ensemble technique will be envisaged for future work, employing the suitable classifiers chosen in this study for preventing attacks in IoT networks.

#### ACKNOWLEDGEMENTS

We would like to thank the members of the University of Kwazulu-Natal's Research office and the open-source software community for the support provided during this research studies.




#### REFERENCES

- [1] A. Aris, S. F. Oktug, and S. B. O. Yalcin, "Internet-of-things security: Denial of service attacks," in *2015 23rd Signal Processing and Communications Applications Conference, SIU 2015 - Proceedings*, May 2015, pp. 903–906, doi: 10.1109/SIU.2015.7129976.
- [2] M. Baykara and R. Das, "A novel hybrid approach for detection of web-based attacks in intrusion detection systems," *International Journal of Computer Networks And Applications*, vol. 4, no. 2, p. 62, Apr. 2017, doi: 10.22247/ijcna/2017/48968.
- [3] B. B. Zarpelão, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, "A survey of intrusion detection in Internet of Things," *Journal of Network and Computer Applications*, vol. 84, pp. 25–37, Apr. 2017, doi: 10.1016/j.jnca.2017.02.009.
- [4] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Computers and Security*, vol. 28, no. 1–2, pp. 18–28, Feb. 2009, doi: 10.1016/j.cose.2008.08.003.
- [5] S. Subbiah, K. S. M. Anbananthan, S. Thangaraj, S. Kannan, and D. Chelliah, "Intrusion detection technique in wireless sensor network using grid search random forest with Boruta feature selection algorithm," *Journal of Communications and Networks*, vol. 24, no. 2, pp. 264–273, Apr. 2022, doi: 10.23919/jcn.2022.000002.
- [6] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Dec. 2006.
- [7] T. Saranya, S. Sridevi, C. Deisy, T. D. Chung, and M. K. A. Khan, "Performance analysis of machine learning algorithms in intrusion detection system: A review," *Procedia Computer Science*, vol. 171, pp. 1251–1260, 2020, doi: 10.1016/j.procs.2020.04.133.
- [8] M. Zaman and C. H. Lung, "Evaluation of machine learning techniques for network intrusion detection," in *IEEE/IFIP Network Operations and Management Symposium: Cognitive Management in a Cyber World, NOMS 2018*, Apr. 2018, pp. 1–5, doi: 10.1109/NOMS.2018.8406212.
- [9] B. A. Tama and K. H. Rhee, "An in-depth experimental study of anomaly detection using gradient boosted machine," *Neural Computing and Applications*, vol. 31, no. 4, pp. 955–965, Jul. 2019, doi: 10.1007/s00521-017-3128-z.
- [10] I. F. Kilincer, F. Ertam, and A. Sengur, "Machine learning methods for cyber security intrusion detection: Datasets and comparative study," *Computer Networks*, vol. 188, p. 107840, Apr. 2021, doi: 10.1016/j.comnet.2021.107840.
- [11] H. Hindy, R. Atkinson, C. Tachtatzis, J. N. Colin, E. Bayne, and X. Bellekens, "Utilising deep learning techniques for effective zero-day attack detection," *Electronics (Switzerland)*, vol. 9, no. 10, pp. 1–16, Oct. 2020, doi: 10.3390/electronics9101684.
- [12] M. B. Pranto, M. H. A. Ratul, M. M. Rahman, I. J. Diya, and Z. Bin Zahir, "Performance of machine learning techniques in anomaly detection with basic feature selection strategy-a network intrusion detection system," *Journal of Advances in Information Technology*, vol. 13, no. 1, pp. 36–44, 2022, doi: 10.12720/jait.13.1.36-44.
- [13] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, pp. 281–305, 2012.
- [14] I. Goodfellow, Y. Bengio, and A. Courville, "Deep learning," *Nature*, vol. 29, no. 7553, pp. 1–73, 2016, [Online]. Available: <http://deeplearning.net/>.
- [15] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Aug. 2016, vol. 13-17-Aug, pp. 785–794, doi: 10.1145/2939672.2939785.
- [16] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001, doi: 10.1214/aos/1013203451.
- [17] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine Learning*, vol. 63, no. 1, pp. 3–42, Mar. 2006, doi: 10.1007/s10994-006-6226-1.
- [18] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.
- [19] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, Aug. 1997, doi: 10.1006/jcss.1997.1504.
- [20] Y. Meidan *et al.*, "N-BaIoT-network-based detection of IoT botnet attacks using deep autoencoders," *IEEE Pervasive Computing*, vol. 17, no. 3, pp. 12–22, Jul. 2018, doi: 10.1109/MPRV.2018.03367731.
- [21] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An Ensemble of Autoencoders for Online Network Intrusion

- Detection,” 2018, doi: 10.14722/ndss.2018.23204.
- [22] H. Kang, D. H. Ahn, G. M. Lee, J. Do Yoo, K. H. Park, and H. K. Kim, “IoT network intrusion dataset.” IEEE Dataport, 2019, doi: 10.21227/q70p-q449.
- [23] I. Ullah and Q. H. Mahmoud, “A scheme for generating a dataset for anomalous activity detection in IoT networks,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12109 LNAI, Springer International Publishing, 2020, pp. 508–520.
- [24] H. A. Afolabi and A. A. Aburas, “Rtl-DL: A hybrid deep learning framework for DDoS attack detection in a big data environment,” *International Journal of Computer Networks and Communications*, vol. 14, no. 6, pp. 51–66, Nov. 2022, doi: 10.5121/ijcnc.2022.14604.
- [25] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *3rd International Conference on Learning Representations*, 2014, pp. 1–15.
- [26] W. J. Conover, “Practical nonparametric statistics,” *John wiley & sons*, vol. 350, 1999.

## BIOGRAPHIES OF AUTHORS



**Hassan A. Afolabi**    received the Bachelor of Science (B.Sc.) degree in computer with electronics from the Lead City University, Ibadan Nigeria and Master of Science (M.Sc.) degree in Computer Science from the University of Ilorin, Nigeria. He is currently studying for his Ph.D. degree in computer engineering at the University of Kwazulu-Natal, Durban South-Africa. He holds several industry standard cybersecurity certifications such as CompTIA security+ and Certified Ethical Hacker (CEH). He is a member of the Nigeria Computer Society (NCS), and the Institute of Information Technology Professionals of South-Africa (IITPSA). His research areas of interest include Machine Learning, Big Data Analysis, Cybersecurity, Deep Learning, and Internet of Things. He can be contacted at email: 219048801@stu.ukzn.ac.za.



**Prof. Dr. Abdurazzag A. Aburas**    received his bachelor's degree in Computer Sciences from Tripoli University, Tripoli-Libya in 1987. He obtained his master's degree in Computer & Information Technology and Ph.D. in Digital Image Processing from Dundee University and De Montfort University, the UK in 1993 and 1997, respectively. He worked in Jordan and UAE universities for five years, Electrical and Computer Engineering Department, Faculty of Engineering, International Islamic University Malaysia, and the International University of Sarajevo. He worked as a visiting full professor at Tecnológico de Monterrey, San Luis Potosi Campus, Mexico. At present, he is working at the University of KwaZulu Natal, School of Engineering, Howard College Campus, Durban, South Africa. He has more than 75 publications at different international conferences and several research papers in international journals. He has research patents in the image processing field. His areas of research interest are Bigdata/Data Science, Machine Learning, Computer Security, Curriculum development, Digital Image Processing, and Human Mobile Interaction. He is a member of IEEE, HCI-UK, ARISE, IITPSA, and IMA Societies. He served as a member board of studies of Engineering Curriculum development (Reviewer and Improvement). He introduced new course curriculums such as Programming for Engineering (2008), Human Mobile Interaction (HMI) (2015), and Bigdata fundamentals (2017). He established Cloud Computing and Mobile Research Group (CCMRG) and Software Engineering Research Group (SERG) (2006-2009). At present, he is the coordinator of the Bigdata research group (UKZNBd). He has supervised several Ph.D. research degrees and MSc research projects. He served as an external examiner for several Ph.D. and MSc. thesis assessors and currently supervising Ph.D. and MSc research projects at UKZN (2019-2022). He has published reference books based on Engineering Education (2013) and Human Mobile Interaction (2015) and his third book on Machine Learning for Engineering using Python in progress. He has more than 20 years of work worldwide in international Universities from 2000 to 2022. He is working in South Africa at UKZN at present time. He was a team leader to build a complete software package for the United Insurance Company in Tripoli-Libya from 1998-1999. The software package is still operating by the company until the present time. He has won several research awards in 2006, 2007, 2008, and 2019. His Research Interests are (but not limited to): Machine Learning, Data Science/Bigdata (Reduction not Compression), Cybersecurity, Digital Signal/Image Processing, Human Mobile Interaction (HMI), and Source Code Energy. He can be contacted at email: aburasa@ukzn.ac.za.